Testing Students' Understanding of Complex Transfer

Andy Way,
School of Computer Applications,
Dublin City University,
Dublin, Ireland.

Email: away@computing.dcu.ie

Abstract

Courses on Machine Translation (MT) need to be tailored to different sets of students with differing skills and demands (Kenny & Way, 2001). Nevertheless, any contemporary course on MT ought to equip students with at least a superficial knowledge of the differences between rule-based and statistical MT; direct and indirect approaches; and transfer-based and interlingual systems. With regard to this latter distinction, the issue of complex transfer is an integral component to this section of a course on MT, whether this be to computational linguists, translators or language students. This paper presents a method of assessing the level of understanding of the issues pertaining to complex transfer for final year undergraduates studying a degree programme in Computational Linguistics. The intention is that this methodology may contribute to a suite of exercises which may be used by other instructors in Machine Translation.

1 Introduction

Courses on Machine Translation (MT) need to be tailored to different sets of students with differing skills and demands (Kenny & Way, 2001). Any contemporary course on MT ought to equip students with at least a superficial knowledge of the differences between rule-based and statistical MT; direct and indirect approaches; and transfer-based and interlingual systems.

With regard to this latter distinction, the issue of complex transfer is an integral component to this section of a course on MT, whether this be to computational linguists, translators or language students. Given the differing backgrounds of these sets of students, the course material, methods of teaching and assessment would all differ, but the notion of how MT systems cope with 'difficult' cases of translation is an integral component of any MT course. These complex transfer cases are much more widespread than is currently perceived: Dorr *et al.* (2002) note that "(cross-language) divergences occurred in approximately 1 out of every 3 sentences" in a corpus of 19K sentences taken from the TREC El Norte Newspaper Corpus.

We teach a course on MT to final year undergraduates studing a degree programme in Computational Linguistics. In addressing the issue of complex transfer, we refer to the excellent presentations of the subject in the standard MT textbooks (Hutchins & Somers, 1992; Arnold *et*

al., 1994; Trujillo, 1999). Given that our students have had 3 years exposure to high-level computing, courses on formal linguistics and computational linguistics as well as tuition in their chosen foreign language (French, German or Spanish), one can see from this profile that these students might be expected to fill industrial positions where they will have considerable influence in establishing which MT systems might be used in the workplace. Furthermore, such students may find themselves in the position of implementing changes to current systems, or indeed developing new ones. Accordingly, it is imperative that they be equipped not just with theoretical knowledge of the various types of complex transfer, but know what the system design implications are for dealing with these examples in practice.

Given the classification scheme in (Trujillo, 1999:124–128), the students are shown how to:

- draw $\langle source, target \rangle$ interface representations which state the dependencies in a translation example in terms of `GOV-ARG-MOD` relations;
- code up solutions to some of these problems in Prolog.

They are made aware of the differences between source and target representations (which, essentially, use 'grammar' rules) and the rules needed to map between such objects (translation rules). They also know that the $\langle source, target \rangle$ dependency structures need to match the LHS and RHS of the translation rules respectively, but also that these latter need to be more general than the specific dependency structures used to represent the various translational phenomena. They also know how default and specific translation rules interact, and that if a specific transfer rule applies, the default rules need to be suppressed in such cases.

2 Assessing Understanding of Complex Transfer

In order to test their knowledge, students are asked to:

1. Find ten instances of complex transfer.
2. Classify each of these cases in terms of the schema given in (Trujillo, 1999).
3. Draw $\langle source, target \rangle$ dependency structures for each sentence.
4. Write a translation rule (using a notation of their choice) capable of mapping the source structure into the target structure for each sentence pair.
5. For one complex transfer case, write a 'Trujillo-style' Prolog rule which takes the input string and generates the target equivalent. They show this by producing a *logfile* demonstrating their programs running successfully.

This assessment is designed to test students' understanding of the different types of complex transfer cases, to see whether they are able to describe them using a formal linguistic notation, and whether they can implement solutions to these cases.

2.1 Drawing Dependency Trees

The students are shown how to draw true dependency trees (cf. Schubert, 1987), but for

the most part, we choose to express dependency relations using a lowered governor representation, as in *Eurotra* (Bech & Nygaard, 1988), but nothing hangs on this.

Assume a thematic (or relation changing) complex transfer case such as:

(1) *DE*: Das Photo ist Hans mißlungen
$\longleftrightarrow$Hans ruined the photo
Lit.: The photo is Hans ruined

The following source and target dependency trees might be drawn:

(2)
```
{cat=s,tense=past,aux=yes}
    ----------------
    |       |      |
   GOV     ARG1   ARG2
    |   {def=yes}  |
    |       |      |
misslingen Photo   Hans
```

(3)
```
{cat=s,tense=past,aux=no}
    ----------------
    |       |      |
   GOV     ARG1   ARG2
    |       |   {def=yes}
    |       |      |
  ruin     Hans  photo
```

These two trees embody everything required to achieve the successful translation in (1): each tree is reduced to PREDs, with all extraneous grammatical information featurized.[1]

[1]Note that we are using a shortcut in the structures for ease of presentation, correlating node identifiers with syntactic roles. It would be better to write instead: `ARG1:{role=arg1}` etc.

2.2 Writing Transfer Rules

Using a notation of their choice, students have to show that a source dependency structure can be mapped into the appropriate target tree by means of a transfer rule. They do this for each of the complex transfer types in the Trujillo classification, and in addition, consider examples where (a) such complex transfer cases co-occur and (b) specific and default structural rules need to interact correctly. Each of these will be discussed in turn.

2.2.1 Argument Switching

Given a source tree (2) and the intended target tree (3), a transfer rule is required to relate the two. This might be formulated as (4):

(4)
```
S:{cat=s}:
  [GOV:{cat=v,role=gov,pred=misslingen},
   ARG1:{cat=np,role=arg1},
   ARG2:{cat=np,role=arg2},
   MODS:*{role=mod}]
=>
S: <GOV,ARG2:{role=arg1},
    ARG1:{role=arg2},MODS>.
```

Again, we write rules using a *Eurotra*-like notation, but any alternative notation may of course be chosen. We represent source and target objects differently (cf. the brackets), as they *are* different objects. The source structure has been produced by the analysis module for that language, so has been *validated* by that grammar. The target structure, meanwhile, has yet to be accepted (the technical term is *consolidated*) by the generation grammar for that language, hence the different bracketing.

Adding `pred=misslingen` on the LHS of the rule constrains the applicability of this rule. We

```
S:{cat=s}: [GOV:{cat=v,role=gov,
                (pred=misslingen;gefallen)},
           ARG1:{cat=np,role=arg1},
           ARG2:{cat=np,role=arg2},
           MODS:*{role=mod}]
=>
S <GOV,ARG2:{role=arg1},ARG1:{role=arg2},MODS>.
```

Figure 1: A more general Transfer Rule for Argument Switching

assume the presence of a set of default rules such as (5):

(5)
```
S:{cat=s}: [GOV:{cat=v},
            ARG1,
            ARG2,
            MODS:*{role=mod}]
=>
S <GOV, ARG1, ARG2, MODS>
```

We can expect a rule such as (5) to translate straightforward transitive verb cases (e.g. *kick* $\longleftrightarrow$*treten*). Unless `pred=misslingen` is added in (4), its LHS will also match *treten* which will result in a mistranslation, as *treten* does *not* require its arguments to be switched.

Note that no `pred` form needs to be inserted on the target side. This approach presumes a lexical entry *misslingen* $\longleftrightarrow$*ruin*. The presence of the `GOV` identifier on the RHS means that apart from the different lexical items (of course), all other source features are merely copied over intact on the target side. However, we need to note explicitly that the two arguments change roles on the target side.

Depending on the complexity of the envisaged system, one might like to generalize rule (4) further, as in Figure 1. That is, both *misslingen* and *gefallen* are argument-switching verbs. The round brackets indicate disjunction. Note that this approach would not work if we reversed the translation relation, as *like* translates both as *gefallen*, in which case its arguments are switched, and *mögen*, in which case they are not. For the former example, this is taken care of by mentioning *gefallen* on the target side (as well as *like* on the source side, of course), while the *like* $\longleftrightarrow$*mögen* case is solved by the default rules in (5).

In order to ensure that the notation is clearly understood by the students, the following summary points are made:

- if a node identifier is mentioned on the target side, it means it is *lexically linked* (by means of a bilingual lexical rule, e.g. *like* $\longleftrightarrow$*mögen*) to its source equivalent;
- if a node identifier appears on the source side, but not on the target, it means that that node and all its structure are *deleted* in translation;
- if a piece of structure appears without a node identifier on the target side, it means that it is *inserted* in translation.

2.2.2 Headswitching

An example of Headswitching from French to English is the following:

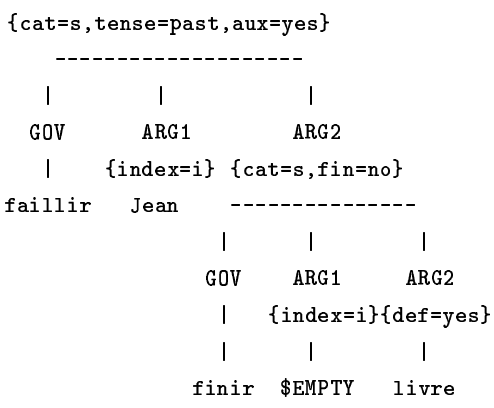
(6) *FR*: Jean a failli finir le livre $\longleftrightarrow$John has almost finished the book
Lit.: John has missed to-finish the book

The source and target dependency structures are those in (7) and (8):

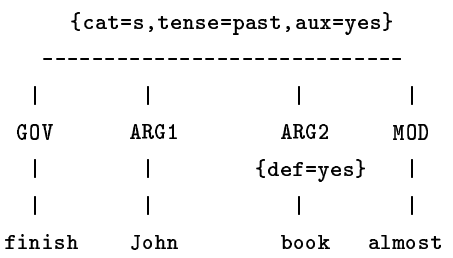
```
S:{cat=s}: [GOV:{cat=v,role=gov,pred=faillir},
            ARG1:{cat=np,role=arg1,index=i},
            ARG2:{cat=s,role=arg2,fin=no}
                    [G2:{cat=v,role=gov},
                     A1:{pred='$EMPTY',index=i},
                     A2:{cat=np,role=arg2},
                     M2:*{role=mod}],
            MODS:*{role=mod}]
=>
S <G2,ARG1,A2,{role=mod,pred=almost},MODS,M2>.
```

Figure 2: A French-English Headswitching Transfer Rule

(7)



(8)



The rule that is required to transfer (7) into (8) is that shown in Figure 2. Again, we have reduced all lexemes to root forms, with appropriate syntactic information encoded as features. Note also the *canonical* (i.e. fixed) order in the dependency structures: it does not matter if a sentential modifier comes at the beginning or the end of a sentence—MODs have a fixed (rightmost) place in the dependency structures. Furthermore, the set of roles is fixed, so we see `role=gov` twice, not `gov1, gov2` etc.

Observe also that subcategorization requirements are fulfilled in dependency structures, whether an argument is overtly realised or not at the level of surface structure. We include a node `M2` in the subordinate clause as modifiers can occur here too, of course. It is the task of the English generation component to ensure that any modifiers (*almost*, and any others under `MODS` or `M2`) appear in the correct order. One might also conceive of an alternative notation which states that *almost* $\in$ `MODS`.

Finally, given that this is a headswitching case, note that the head in French is *faillir*, identified as the node `GOV`, while in English the head governor is the head of the embedded phrase in the source structure, namely `G2`. The source nodes identified by `GOV`, `ARG2` and `A2` do not appear on the RHS of the rule, meaning that they are deleted in translation. Some students write `GOV:{role=mod,pred=almost}`, which presupposes that *faillir* and *almost* are linked in the lexicon. This would be wrong, so the only way to relate them is structurally, i.e. in a rule like that in Figure 2. `fin=no` says that the sentential `arg2` is non-finite ('finiteness equals no').

2.2.3 Other Complex Transfer Cases

The students are asked to provide similar descriptions of cases of Structural Divergence, Lexical Gaps, differences in Lexicalization, Categorial changes, Collocational differences, and Idioms. For reasons of space, we omit any further details of these examples here.

2.2.4 Implementing 'Trujillo-style' Prolog rules

Having shown the students in class how some simple cases can be handled by using 'Trujillo-style' Prolog rules, they are asked to write such a rule for a complex transfer case. This might be as in (9) for the *like* $\Longrightarrow$*gefallen* relation-changing case between English and German:

(9)
```
:- op(500,xfx, '<==>').
:- op(450,xfx, dtrs).

[s] dtrs [NPsE,
         [vp] dtrs [[v] dtrs [like],
                            NPoE ] ] <==>
[s] dtrs [NPsD,
         [vp] dtrs [[v] dtrs [gefallen],
                            NPoD ] ]  :-
   NPsE <==> NPoD,
   NPoE <==> NPsD.

i <==> mir.
you <==> du.
```

When tested in Prolog, given the input `[s]dtrs[i, [vp]dtrs[[v]dtrs[like], you]`, Prolog returns `[s]dtrs[du, [vp]dtrs[[v]dtrs[gefallen], mir]]`, as desired.

2.3 Co-occurrence of 'hard' cases

The students are also asked to consider sentences where more than one complex transfer case co-occurs. Habash & Dorr (2002) observed the following case in their analysis of the TREC El Norte Corpus:

(10) Maria tiene gustos de políticos diferentes $\Longrightarrow$Different politicians please Maria.

Habash & Dorr point out that there are *four* translation divergence types present in (10): categorial ($gusto_{noun}$ to $please_{verb}$), conflational (*tener gusto* to *please*), thematic (*Maria* and *politicians* switch argument roles), and structural (*políticos* is a modifier of *gustos* in Spanish but *politician* is an argument of the verb in English).

For German, we might embed one headswitching case inside another. Such cases might be *anscheinend* and *zufällig*, as in:

(11) Maria schläft anscheinend $\Longrightarrow$Mary seems to sleep
Maria schläft zufällig $\Longrightarrow$Mary happens to sleep

We can combine these two headswitching cases together, as in (12):

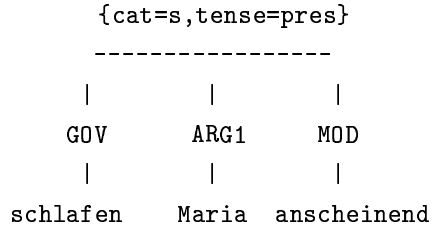
(12) Maria schläft anscheinend zufällig $\Longrightarrow$Mary seems to happen to sleep
Lit.: Mary sleeps seemingly by-chance

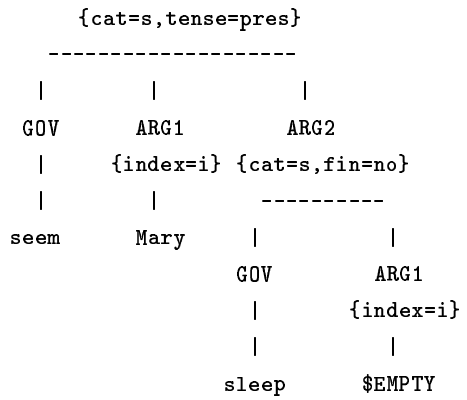
That is, Mary just nods off rather than planning to sleep.

The question which is posed to the students is whether the rules they had written for the *individual* instances of these hard cases cope with this instance where they co-occur. Either answer is fine, of course, as long as they can explain what is going on. However, given the rule writing scheme chosen here, the usual response

is that a new rule needs to be written for the co-occurrence of the two complex transfer cases. As (10) shows, any number of such cases can co-occur, so any approach that requires us to write a new rule *just* for the case where two particular cases occur together is doomed to failure as a general solution to the problems of translation (cf. Way *et al.* (1997), who point out that the Eurotra *EF* and *CAT* formalisms both suffer from this problem).

Returning to our translation example (12), where two headswitching cases co-occur, the structures required for the simpler translation *Maria schläft anscheinend* $\longleftrightarrow$*Mary seems to sleep* are:

(13)


(14)


This requires the transfer rule in Figure 3.

For our more complex example (12), the source dependency structure does match the LHS of Figure 3. So far, so good. What about the RHS? Everything is straightforward till we get to the translation of *zufällig*. On the LHS, this is identified as `MODS`, but on the RHS we see

```
S:{cat=s}:[GOV:{cat=v,role=gov},
           ARG1:{cat=np,role=arg1},
           MOD: {role=mod,pred=anscheinend}],
           MODS:*{role=mod}]
=>
S <{role=gov,pred=seem},
   ARG1:{index=i},
   {cat=s,role=arg2,fin=no}:
             <GOV,
              {pred='$EMPTY',index=i},
              *{role=mod}>,
   MODS>.
```

Figure 3: A German-English Headswitching Transfer Rule

that this gets translated by default, so we'd end up with the different translation *Mary seems to sleep by chance*. If there had been no lexical rule *zufällig* $\longleftrightarrow$*by chance*, translation would have failed. For instance, compare what would have happened if *gerne* replaces *zufällig* in (12)? This can only be translated structurally, and the RHS of Figure 3 will not be able to cope.

2.3.1 Impact of the Default Rules

The students are also asked to address the following problem: as well as a specific rule like (4) which correctly enables the translation in (1), we also have rules like (5) which will translate *Das Photo ist Hans mißlungen* as *The photo ruined Hans*. How can we prevent this unwanted, wrong compositional translation?

Some of the students came up with nice solutions using the 'Trujillo-style' Prolog implementations. With a set of default rules and specific rules, allowing Prolog to backtrack produced both translations, one right, one wrong. Ordering the specific *before* the default rules, and adding a cut to the specific rule, prevented

backtracking and produced just one translation, the correct one via the specific rule, as required.

3 Concluding Remarks

Irrespective of the intended audience, no undergraduate or postgraduate course in MT can omit the teaching of complex transfer. We have presented here one method by which the understanding of final year computational linguistics students studying MT can be assessed in this important area. As such, this paper will primarily be of interest for those teaching more advanced programming rather than teachers of courses in MT to translators or language students.

Students firstly are asked to find a number of complex transfer cases themselves, and classify them according to the schema presented in Trujillo (1999:124–128). They then have to draw appropriate ⟨*source, target*⟩ dependency trees for each translation example, and write a transfer rule capable of mapping between the two dependency structures. They implement a solution to one of these cases in Prolog using a 'Trujillo-style' rule, and finally consider examples where complex transfer cases co-occur and where default translation rules need to be suppressed where a more specific structural transfer applies.

Finally, there are many teachers of MT courses, all of whom have to continually strive to set a number of assessments each year to test students' understanding of the course material. It can prove difficult to come up with new exercises year after year. This author benefitted from attending the Workshop on Teaching Machine Translation at MT-Summit VIII, in that an exercise presented by Pérez-Ortiz and Forcada (2001) has been successfully carried out on our students in Dublin. Accordingly, it is our intention that the methodology presented here might serve as a useful contribution to a suite of exercises which may be used by other instructors in Machine Translation.

References

[1] Arnold, D.J., L. Balkan, R.L. Humphreys, S. Meijer and L. Sadler (1994): *Machine Translation: An Introductory Guide*, Blackwell, Cambridge, Ma./Oxford.

[2] Bech, A. and A. Nygaard (1988): 'The E–Framework: A Formalism for Natural Language Processing', in COLING: *Proceedings of the 12th International Conference on Computational Linguistics*, Budapest, pp.36–39.

[3] Dorr, B., L. Pearl, R. Wha and N. Habash (2002): 'DUSTer: A Method for Unraveling Cross-Language Divergences for Statistical Word-Level Alignment', in S. Richardson (ed.) *Machine Translation: From Research to Real Users*, Springer, Berlin/Heidelberg, pp.31–43.

[4] Habash, N. and B. Dorr (2002): 'Handling Translation Divergences: Combining Statistical and Symbolic Techniques in Generation-Heavy Machine Translation', in S. Richardson (ed.) *Machine Translation: From Research to Real Users*, Springer, Berlin/Heidelberg, pp.84–93.

[5] Hutchins, J.W. and H.L. Somers (1992): *An Introduction to Machine Translation*, Academic Press, London/San Diego.

[6] Kenny, D. and A. Way (2001): 'Teaching Machine Translation & Translation Technology: A Contrastive Study', in M. Forcada, J-A. Pérez-Ortiz & D. Lewis (eds) *Proceedings of the MT Summit Workshop on Teaching Machine Translation*, Santiago de Compostela, Spain, pp.13–17.

[7] Pérez-Ortiz, J-A. and M. Forcada (2001): 'Discovering Machine Translation Strategies Beyond Word-for-Word Translation: a Laboratory Assignment', in M. Forcada, J-A. Pérez-Ortiz & D. Lewis (eds) *Proceedings of the Workshop on Teaching Machine Translation*, MT-Summit VIII, Santiago de Compostela, Spain, pp.57–60.

[8] Schubert, K. (1987): *Metataxis: contrastive dependency syntax for machine translation*, Foris, Dordrecht.

[9] Trujillo, A. (1999): *Translation Engines: Techniques for Machine Translation*, Springer, London.

[10] Way, A., I. Crookston and J. Shelton (1997): 'A Typology of Translation Problems for Eurotra Translation Machines', *Machine Translation* **12**:323–374.